DETERMINING POINT-OF-COMPROMISE

BACKGROUND

TECHNICAL FIELD

[0001] This disclosure relates generally to computer data storage, data mining and knowledge discovery. More particularly, the disclosure relates to determining relationships in order to find a probable point-of-compromise from a correlated database of interactions of members therein from patterns and relationships in the data.

DESCRIPTION OF RELATED ART

[0002] Along with the processing power of modern computers has come the amassing of digital data into databases having sizes which tax reasonable data storage limits and computing power, the ability to derive meaningful conclusions from the data in a prompt, or "real-time," manner. Fast computer memory such as random access memory integrated circuits are relatively expensive for use as massive databases; the use of mass data storage apparatus, such as electromagnetic tape or electro-optical disk drives, are far more economical but have attendant significant compromises with respect to the access time to data. Yet within such massive databases lies information which can be of strategic importance.

[0003] For example, consider the simple problem of credit card,

and debit card, fraud. A loan service or clearinghouse service company, such as Discover[tm], Mastercard[tm], Visa[tm], or the like, licenses its logo to a plurality of lending institutions which then issue individual credit cards; each transaction processed is funneled through the company. A miscreant employee occasionally steals customer credit card numbers and identification and later sells them on the Internet. A cloned card may complete several fraudulent transactions before the problem is recognized and the card is cancelled. With the enormous number of worldwide authorized merchants, credit cards, and credit card transactions per day, determining where the originating card theft likely occurred, the point-of-compromise, is a seemingly impossible task. Even repeated offenses at a specific merchant site are unlikely to be discovered easily or quickly, because the card numbers stolen are distributed among different card issuers and no fraudulent transactions are necessarily committed at the actual point-of-compromise. By avoiding any simple patterns like stealing only "Ajax Credit Cards," the thief hopes to thwart each individual issuer.

[0004] With there being an enormous number of credit card payment transactions on each issuer's cards on a daily, or even hourly, basis, tracking each transaction and attempting to

determine a probably point-of-compromise from bogus transactions is likely an impossible task without the aid of computer data storage and data mining. Yet even with modern computing power, there are problems. For example, given a worldwide set of ten million merchants and a set of five hundred million credit card holders, to log a simple correlation of each credit card use to each merchant identification in a brute-force dense format, e.g., where one bit in a two-dimensional memory matrix designates a transaction, would require approximately 568 terabytes - - 568,000,000,000,000 bytes - - of storage. Thus, the data compilation task for a single credit card company becomes unwieldy.

[0005] Moreover, the point-of-compromise may not even be a particular merchant's employee. For example, merchants at a mall may combine computer terminal operations through a transaction aggregation server for the entire mall provided by each credit company, card issuer or conglomerate of issuers. That is, each checkout register is merely a terminal for a mainframe computer located somewhere other than at the specific merchant given an individual identifier. A hacker may compromise the mainframe and steal credit card information. Similarly, the problem is exacerbated greatly by Internet commerce which relies heavily on credit transactions and where

a Web site, such as that of a merchant or an escrow service, can be compromised by a hacker.

[0006] Another example of a need for determining possible point-of-compromise would be for products which may see a plurality of uses and users over time, e.g., portable medical equipment.

[0007] Still another example for point-of-compromise determination problem relates to original equipment manufacturers having worldwide mass distribution channels, such as integrated circuit "chip" manufacturers. A manufacturer may have ten machines fabricating hundreds of thousands or even millions of chips of a particular design which are then distributed all over the world via an appliance manufacturer having a global installed base, e.g., chips in cellular telephones. If at some later time a significant chip hardware problem arises in the installed base, the manufacturer may need to determine which of the ten machines and which particular runs producing the potentially defective chips still in service are affected in order to contact users such as for a for a recall. Chip serial numbers must be correlated to machines and particular time-stamped runs for each chip yield of the machines to find a potential point-of-compromise. Other mass market distributed items of manufacture having user-critical characteristics, like

other computer-related consumables such as ink-jet cartridges, sterile medical and surgical supplies, sealed-package foods, and the like, produced on a variety of machines at a plurality of locations may carry the same problem of identification of the specific source of a defect.

[0008] In the state of the art, the only practical and economical data storage is mass data storage apparatus. Where time is of the essence, for example where a credit issuer is sustaining bogus credit card transactions hourly with limited liability for reimbursement from the card holder, fraud costs significantly affect the cost of doing business. Yet access to the data may mean serially searching a library-like storage room filled with back-up data tapes of serially-logged transaction records. Finding every transaction for a specific card over a specific time frame prior to the start of fraudulent use in order to try to identify the origination of the fraud - - e.g., the merchant having the felonious employee - - is like finding a veritable needle in the haystack. Clearly, the situation worsens with the unknown factor of time delay the thief may have used to mask the initial time of each theft. It can thus be recognized that rapid access to meaningful information in such massive data bases within a relatively rapid time frame is an important and significant task.

[0009] There is a need for solving such data mining, knowledge

discovery tasks with minimal processing power and data storage requirements.

BRIEF SUMMARY

[0010] The basic aspects of the invention generally provide for a data storage and data mining for detecting a possible point-of-compromise.

[0011] Aspects of the invention are described with respect to exemplary embodiments associated with the problem of credit card fraud.

[0012] The foregoing summary is not intended to be inclusive of all aspects, objects, advantages and features of the present invention nor should any limitation on the scope of the invention be implied therefrom. This Brief Summary is provided in accordance with the mandate of 37 C.F.R. 1.73 and M.P.E.P. 608.01(d) merely to apprise the public, and more especially those interested in the particular art to which the invention relates, of the nature of the invention in order to be of assistance in aiding ready understanding of the patent in future searches.

(7) BRIEF DESCRIPTION OF THE DRAWINGS

[0013] FIGURE 1 in accordance with an exemplary embodiment of the present invention is a chart illustrating a matrix structure for logging transactions.

[0014] FIGURE 2 is a flow diagram illustrating storing of data relevant to point-of-compromise in accordance with an exemplary embodiment of the present invention as shown in the embodiment of FIGURE 1.

[0015] FIGURE 3 is a flow diagram illustrating a specific exemplary embodiment for implementing point-of-compromise determination in accordance with the embodiments of FIGURES 1 and 2.

[0016] FIGURE 4 is a flow diagram illustrating a generic implementation for point-of-compromise data storage and data mining in accordance with the embodiments of FIGURES 1 and 2.

[0017] Like reference designations represent like features throughout the drawings. The drawings in this specification should be understood as not being drawn to scale unless specifically annotated as such.

(8) DETAILED DESCRIPTION

[0018] The present invention is described herein by continuing the Background section description of credit card fraud as a significant exemplary implementation. This is merely to facilitate understanding of the invention which is applicable to any commercial transaction or industrial operation situations where a very huge number of individual events need to be logged into

a useful database; no limitation on the scope of the invention is intended by the inventors nor should any be implied therefrom. The scope of the invention is set forth by the claims hereinbelow.

[0019] Turning now to **FIGURE 1**, assume as an example that a fictitious company, Ajax Credit Company, licenses a plurality of banks, or other issuers, to issue a million credit cards, "$CC_{1 \text{ through } H}$," where "H" = last cardholder identifier, each credit card having a unique alphanumeric-like identifier, e.g., a sixteen digit number. Ajax signs on several million merchants, "$M_{1 \text{ through } S}$," where "S" is the last authorized Seller, to accept the Ajax logo'd credit cards for transaction payment. A matrix of card identifiers versus merchants can be formed by Ajax, or its designated agent or associated business for tracking individual card usage. Clearly, the number of cards issued and accepted and the number of authorized merchants can change significantly on a monthly, weekly, daily, or other given time frame. The company can decide a specific significant time frame for logging data based on the particular implementation criteria involved.

[0020] It will be recognized by those skilled in the art that for each transaction the stored data can be minimized. A pair of identifiers, (MERCHANT, CARD) in seven bytes of data suffices, where four bytes are enough to address 4.3 billion

credit card holders (30 bits per billion) and three bytes are enough to address 16 million merchants.

[0021] To minimize memory requirements, the minimal amount of information is used to construct a matrix 100. As shown in FIGURE 1, a simple two-dimensional merchant, card - - M_i, C_i - - data pair matrix is used to record each transaction. Note that for other implementations more complex three-plus dimensional matrices may be employed. Each transaction between a cardholder and a merchant is correlated with a one-bit 101 in the matrix 100; e.g., a digital "1" at memory location M_1,CC_2, 101 signifies a transaction, or "hit." Computer automation of filling "hits" in the matrix for each predetermined time period would be known to those skilled in the art.

[0022] Given such a matrix, or more likely for example, a set of daily or weekly matrices that cover a set of time frames, the computer can scan for each given compromised credit card number and extract those merchants having had a transaction, whether bona fide or bogus, with each of said credit cards. Note that the date-range employed for each card number may vary. That is, e.g., one card may have been compromised a year ago and another card may have been compromised yesterday; the scan date-range of the past, or "past window," for each card is tailored accordingly with respect to each date first

known that the card has been compromised, at some unknown time in the past.

[0023] Referring specifically to FIGURE 1, assume credit cards CC_2, CC_3 and CC_{H-1} are reported, or otherwise determined, as being compromised. Scanning down the column 103 for compromised card CC_2 would extract merchants M_3 and M_{S-1}. Scanning down the column 105 for compromised card CC_3 would extract merchants M_2, M_3 and M_s. Scanning down the column 107 for compromised card CC_{H-1} would extract merchants M_1, M_3 and M_s. The "hit" score for each listed merchant is tallied:

$$M_1 = 1$$

$$M_2 = 1$$

$$M_3 = 3$$

$$M_{s-1} = 1$$

$$M_s = 2.$$

The score being the highest for merchant M_3 rates the highest probability for being a probable point-of-compromise for the given set of compromised credit cards; merchant M_s rates the next highest probability, and merchants $M_{1, 2, s-1}$ rate the lowest probability. In other words, since it is likely that a malfeasant employee has compromised a plurality of credit cards, finding the merchant where the most number of compromised cards

have been used is a good predictor of where the felonious activity has occurred.

[0024] Many known manner matrix database building and manipulation commercial products from Microsoft, Oracle, Sybase, and the like are known in the art. Further description is not necessary for an understanding of the present invention. Moreover, many mathematical abstraction techniques are known in computer science for analyzing statistical data, e.g., matrix, sparse matrix, Logfile, B-Tree, and the like may be adapted for data analysis, and further description is not necessary herein.

[0025] Keeping and manipulating a full matrix where many millions of authorized merchants may be serviced by the credit card issuer and many times more millions of cards may have been issued, keeping a complete matrix on even a daily basis would have very large hardware storage requirements. Furthermore, an enormous table of information showing data related to every transaction is inefficient to process. Thus, in the preferred embodiment, in order to reduce the memory and data retrieval requirements, hashing the data (using representations of data, also referred to as "keys," to store and retrieve data) using modulo N arithmetic is employed where N=256 when creating time logged files. For example, in the

logging computer, on a daily basis, 256 separate files are opened, wherein each file contains a portion of the entire days logged transactions. It will be recognized by those skilled in the art that technical progress and affordability in technology computer storage and data processing would allow 256^n files, greatly increasing the power of the algorithm described herein. More specifically as an example, the credit card number columns in FIGURE 1 are divided into 256 groups. All transaction bits which belong to the first group are logged in the first file, all transaction bits which belong to the second group are logged in the second file, and so on for all 256 groups. Thus a resulting collection of identified objects, namely transaction "hits" in a specific group have the same type of identifying structure. In other words, using modulo N arithmetic, a file may contain all transactions related by, for example, the last digit of a plurality of cards, namely all issued cards with their 16-digit numbers ending in 3 "$C_{...3}$" that had at least one authorized merchant transaction logged on that day. Moreover, the "3" need not be stored repeatedly in the file, also saving storage space and processing time. As will be recognized by those skilled in the art, a variety of such hashing techniques, parallel processing, and other known manner data storage techniques may be employed for organizing such an intrinsically

large database into a manageable files and optimizing computing power.

[0026] **FIGURE 2** is a process 200 flow diagram illustrating the applicable storing of data in accordance with a specific exemplary embodiment of the present invention related to a credit card database and credit card fraud. As mentioned above, a brute-force matrix storage and scanning for every hit can be implemented, but only with great disadvantages as to required storage space and microprocessor input-output functions. Therefore, data storage and processing time should be optimized.

[0027] Assume that the credit card issuer Ajax keeps daily log files for each transaction involving an issued Ajax credit card. In accordance with the preferred embodiment, for each twenty-four hour period - - e.g., starting 12:00 a.m. Greenwich Mean Time ("GMT") - - Ajax, or his agent such as a business associate assigned the task of tracking credit card fraud, creates 256 new files, 201. The issuer then waits, 203, to receive data for each transaction. Each transaction will be received, 205, and include the credit card number and a merchant identification where each merchant had received a unique identifier, "M_S."

[0028] Here it is useful to consider the situations described in the Background section where issues in the installed-base field -

- such as merchants with like names, merchants such as large department stores with hundreds of terminals, merchants changing computer systems or aggregation services, many Web sites belonging to a single merchant, or the like -- can create misdirection during data analysis or at least less accurate data for pinpointing a point-of-compromise. Each entity merchant, terminal, aggregator, or the like, preferably is tracked; each may therefore constitute another row, or column, of a credit card transaction data storage matrix. It is also preferred that instead of logging just merchant numbers, that an abstraction of "Situation Identifiers" ("SID") may be established; in other words, to reduce storage and process requirements, a unique SID is assigned to multiple entities which are related. For example, given SID_{203}=NJPMAT48 may represent the physical "New Jersey Paramus Mall Aggregator:Terminal 48" which is known to be in Macy's Department Store in that same mall. Note also, that given further information about SIDs established for entities originally, e.g., Borders Books NYC, and Borders Books 10002, later being determined to be the exact same physical location store entity, may allow for condensing data and future transactional data. In effect, multiplexing entities with an SID permits updating cumulative data storage in a simpler manner than trying to update a simple, linear, transaction-by-transaction

data listing. Each change to the macro-system requires assigning a new SID in order to track the different configuration separately. Then, at time of tally, each SID extracted can be correlated to one or more related merchants, aggregations, computer terminals, or the like, implicated. It is preferred that each transaction include a relational data pair structured as: (SID_i, CC_H) and a separate database correlating SIDs to a list of entries.

[0029] Preferably, each data pair is manipulated 207 to compress storage and sorting requirements. Let the last byte be used to group credit cards $CC_{...0}$, $CC_{...1}$, $CC_{...3}$ through $CC_{...9}$ for purposes of data sorting. That is, a file is created for all cards ending in "0000 0000" binary, another file for all cards ending in "0000 0001" binary, et seq. For example, as a credit card generally has a unique 16-digit number, storing each complete number is memory extravagant. The last digit, 8-bits, lowest order byte of the card number is split out into an 8-bit variable file number and the remaining bits are a "rest" variable. In a known manner, the notation is thus:

$$(fileno_i, rest_i) = split\ (cardno_i).$$

For data storage purposes, for each specific transaction a data pair $(rest_i, SID_i)$ is appended 209 to the file number and stored accordingly in the appropriate one of the 256 opened files.

[0030] At the last recordable transaction of the logging day, e.g., it is now 12:00 a.m. GMT of the next day, 211, YES-path, the files are closed 213 and identifiably marked for retrieval accordingly, such as in a known manner of date/time-stamping. If the last transaction was not the last recordable transaction of the day, the process loops to the wait state 203.

[0031] Appending to files of a database is well known in the art, commonly referred to as opening files in an "append mode." New files or new databases may be created based on other time periods - - e.g., closing a database at the end of each day, month, or other given time period. When analyzing the data later, many files are eliminated as irrelevant to particular compromises simply by their date information. Therefore, maintaining a time-based storage of all interactions may be implemented in accordance with the needs of any specific implementation. Generally then, in such a manner, Ajax has a set of manageable files and can find relevant, segregable, time windows, from what is likely to be millions of actual past transactions. Each file is associated with a common structure or characteristic of the matrix-resident identifier, such as a last digit or data byte for issued credit cards. At some later time then, namely, once a compromised card number is known, only the files associated with that structure or characteristic need be

searched and the associated data pair identifiers, namely the correlated SIDs, need be retrieved and tallied. Again, from their tally score, a point-of-compromise origin can be predicted.

[0032] **FIGURE 3** is an exemplary embodiment for implementing the point-of-compromise estimation procedure 300 generically. At some point in time it becomes apparent to the user that a system-of-interest - - the Ajax Credit Co. system, the installed base of defibrillators, the identification code for a set of consumable products, or the like - - has been compromised at an unknown fraud origination point, or points, involving more than one specific item.

[0033] Continuing the Ajax Credit Co. credit card system exemplary embodiment, assume that several credit card numbers have been stolen; therefore, there is a set of known unique compromised credit card numbers 301. For each specific compromised credit card number, $cardno_i$, there is a related file 303 as described with respect to FIGURE 2.

[0034] Ajax can collect, or maintain, a relational list of credit card numbers, "cardno," for each file number, "fileno." Files not associated with a compromised credit card number are immediately eliminated from being a suspected point-of-compromise since there will be no relevant transaction data in those files. Thus, the Ajax Credit Co. quickly extracts only those

files related, namely those files having one or more compromised card numbers. In other words, from a collected list of credit card numbers for each file, those files having a non-empty list with respect to the set of compromised cards, or files true "hits," are now the only files-of-interest. Each relevant extracted file (if coincidently all compromised card numbers are in a single file) or files 305 will be separately considered serially or iteratively

[0035] By definition, each transaction will have a given timestamp. Only a predetermined date range will be of interest to Ajax. For example, the first compromised card transaction may have occurred yesterday, a second compromised card two months ago, or all of the compromised cards currently under consideration may have been reported only within the last week. However, if related, the actual point-of-compromise problem may have occurred previously; e.g., a malevolent employee may have begun stock-piling stolen credit card numbers months in advance of his sale of them. Therefore, the user should want to use a predetermined date range much greater than just going back to the first date of compromise, e.g., going back to look at all transactions of each compromised card for one year prior to that particular cards first known compromise date. The further back in time one searches, the more likely the tabulation will

cover the point-of-compromise; a reasonable range-of-interest will be dependent upon the specific implementations predetermined data domain knowledge. Therefore, each file 303 is opened 309 that has its file number, "fileno," and its date with the range-of-interest 307.

[0036] Each opened file 305 will contain many more records of card transactions than those associated with the compromised cards 301. Therefore, it is necessary to extract only matching "hits" for the compromised cards 301. This may be done by looping through the files recording the (rest, SID) specific transaction pairs 311 described hereinabove and making a determination 313 whether the $rest_i$ matches any $rest_i$ in the comprised set 301, 305. Each pair in the open file is scanned 315, 313. For each match 313, YES-path, each entity, for the purpose of a compromised credit card an exemplary authorized merchant entity (M_s, FIGURE 1) is identified 317. In other words, a compromised card on the list has been correlated to a specific merchant and a specific transaction of the virtual matrix. Therefore, it is appropriate to keep a relational score as described above. That merchant is assigned to the compromised card "hit" tracking dataset; the tally for that merchant is incremented 319 for each "hit" in each SID for the current card, "rest." A check 321 is made for other "hits" on the

same merchant (e.g., at a different register in the same store on the same date) and compromised card number and the tally incremented appropriately. If there are more relevant files 323, NO-path - - namely if there are more files associated with compromised cards in the set-of-interest 301, 303, 305 and in the date-range-of- interest 307 - - the next file is opened 309. Then, looking to decision block 321, YES-path, the iteration loop 315, following the NO-path, 311, 313, 317, 319, 321 is repeated until the last record - - that is the last (rest, SID) pair of the last opened file - - is analyzed, 323, YES-path. The process repeats for each compromised card. Programmers skilled in the art will recognize the efficiency benefits of performing these scans for all compromised cards at once, described hereinafter.

[0037] Once the last relevant file of the last compromised card has been analyzed 323, YES-path, merchant tallies can be compiled 325. An output 327 based upon the tallies is provided. The output 325 may be structured to fit any particular implementation. A tally descending-count ordered list would be a simple output based on the assumption that the merchant with the highest score has the highest probability of being the point-of-compromise.

[0038] There are a variety of data adjustment options which may be employed as part of compiling merchant tallies 325. For

example, in order to better represent a final probability of point-of-compromise, normalization of each score, tally, can be implemented. For example, each tally may be normalized by the quantity of business each merchant transacts. Each tally entry is divided by the activity level of the merchant for the predetermined time period range-of-interest. Another exemplary factor may be merchant sales volume. It will recognized by those skilled in the art that such normalization, or other adjustment factors, for the data may be employed based upon the specific implementation and its related data domain knowledge.

[0039] Moreover, the tally compilation analysis may employ machine learning options. The tally feature of the process 300 may be used as an input which can feed back to and modify future compilation analyses. For example, given information of which merchants were found out to be an actual point-of-compromise by a prior analysis process 300, a labeled training set of merchants can be established. Known manner machine learning algorithms can use this unadjusted tally set as one potential predictive feature among others, such as transaction totals, sales volumes, and the like. A known manner machine learning component can then be trained to recognize merchants that match the determined pattern. This classifier, preferably

providing a probability output, can be used, given the tally, to estimate which merchants are most likely points-of-compromise.

[0040] Other options are available for further refinement of the process 300. Each log period, e.g., each day, the matrix is updated as to correlated transactions. However, over time, merchants may merge, merchant identification codes may be recognized by Ajax Credit Co. as having been mistaken as distinct merchants but are actually a single entity, and the like. A relational adjunct database, e.g., an organization tree, can be maintained for referencing such new data correlating factors as each becomes recognized. The table is used when determining entities from SIDs.

[0041] Similarly, rather than storing individual merchant identifiers, M_s, in the preferred embodiments described above, an abstraction such as the transactions situation identification, SID, was employed. As part of the tally subprocess 325, a table can translate the abstraction back to a current definition of the merchant-of-interest. Storage of "hit" matrix data can then be reduced to only requiring the individual SIDs rather than each and every possible entity identifier, M.

[0042] As another option, note that once a tallying is established, at-risk credit cards may be predicted. A second indexed dataset, sorted by merchants having a relatively high

probability of being a point-of-compromise can direct the issuer with respect to which Ajax credit cards should be the subject of an intensified watch for future fraudulent uses. In other words, given a list of most likely point-of-compromise merchants, the output can be a list of credit cards sorted by their score which can be traced to most likely at-risk customers. This may be used for example to notify such customers or to notify a plurality of other merchants to double-check identification on specific cards having an attempted use during a given time period.

[0043] As another option, a predetermined probability score can be associated with each merchant. When a merchant accepts a credit card and electronically transmits the credit card number for a dollar limit approval, a calculation can be quickly computed to determine the likelihood that the card is compromised. For example, the merchant may have a score normalized by transaction volume to represent a probability that a given credit card number is compromised. It is possible to either (1) simply filter the credit card list (FIGURE 1) by a predetermined probability threshold, only selecting those merchants above a certain threshold, or (2) instead of tallying one full count for each cart transaction, accumulate the probability that the card has not been compromised. For example, this optional process would initialize the output scores to 1.0 for each credit card. When

tallying a transaction between a given credit card, e.g., C_{H-1}, and a given merchant, M_1 who has a probability of compromising the card of 1%, multiplying the output probability by (100-1%=99%) determines the posterior probability that the card has not be compromised. The cards with the smallest output probability are the ones that are least likely not compromised and therefore represent the most at-risk transactions. Again, the merchant can be notified of the risk in the current transaction and security precautions can be implemented in real-time.

[0044] **FIGURE 4** represents the process described hereinbefore in a generic manner suited to computerized implementation. It assumes a company has maintained files for each identifiable item - - in this exemplary embodiment still using credit cards as items being tracked for each use thereof. A computer program in accordance with this algorithm can be commercialized via known manner commercial distribution to be used by a credit processing company or its business agent(s).

[0045] The input 400 for this data mining, knowledge discovery, processing algorithm is a set of compromised identifiable items - - e.g., credit card numbers - - wherein each has a suspected date window - - a search parameter defining a reasonable past time period before a given time-of-first-known-fraud. The term "card" and "product" or "equipment" and the like, or simply

"item," are thus interchangeable depending on the implementation design goals.

[0046] A set of file numbers in which there will be found compromised items is determined 403. Those files extracted for analysis of the contained data, namely a matrix of transactions for a given time period.

[0047] For each extracted file "f" matching a determined file number and for one or more corresponding suspected date windows 405, and for each (rest, SID) pair, supra, in the file 407, the "rest" is compared to any compromised card number and its date window 409.

[0048] That is, the first extracted file, "f1," first record, "rest $_1$, SID_1," is analyzed. If there is a match, 409, YES-path, a tally is started, or incremented appropriately, for the associated SID.

[0049] The next 413 (rest, SID) pair 407 is then compared, tallying 411 only those SIDs where a match 409, YES-path, is determined.

[0050] When there are no more matches, 409, NO-path, in the current file, "f1," then the next compromised card is similarly analyzed 407, 409, 411, 413.

[0051] When all of the compromised cards have been checked, the next file is opened 415, 405 and the process 407, 409, 411, 413, 415 is repeated.

[0052] Once all compromised cards in their respective date windows have been processed, the tally for each SID is output 417. As with the foregoing detailed description with respect to FIGURES 1, 2 and 3, the score for each SID is an indicator of likelihood of point-of-compromise.

[0053] Note that with this generic process, many techniques for data processing and filtering of data files may be employed. The storage saving by card number data compression has been described above is only one such technique. Similarly, other factors for normalization and indexing may be employed.

[0054] As described hereinabove with respect to various embodiments, the present invention thus relates to data mining and knowledge discovery techniques. The technique may be used by an individual entity or a group of entities organized for the purpose. A data base is established in a virtual matrix form for each transaction of large scale transactional events. Data is filed and logged in a manner for rapid sorting such that given a set of identifiers for comprised transactions, a limited subset of the matrix only need be accessed for specific knowledge discovery in the nature of point-of-compromise. For each potential point -of-compromise, a tally is compiled. Potential points-of-compromise are sorted according to tally score. Score is indicative of increasing likelihood of a source point-of-

compromise.

[0055] The foregoing Detailed Description of exemplary and preferred embodiments is presented for purposes of illustration and disclosure in accordance with the requirements of the law. It is not intended to be exhaustive nor to limit the invention to the precise form(s) described, but only to enable others skilled in the art to understand how the invention may be suited for a particular use or implementation. The possibility of modifications and variations will be apparent to practitioners skilled in the art. Credit card transactions, debit card transactions, or other large scale transactional payment surrogates, mass piece part manufacture with global distribution, consumables such as sterile medical and surgical supplies, and the like, all can be adapted to implementations of the present invention for data mining and knowledge discovery. No limitation is intended by the description of exemplary embodiments which may have included tolerances, feature dimensions, specific operating conditions, engineering specifications, or the like, and which may vary between implementations or with changes to the state of the art, and no limitation should be implied therefrom. Applicant has made this disclosure with respect to the current state of the art, but also contemplates advancements during the term of the patent, and

that adaptations in the future may take into consideration those advancements, in other word adaptations in accordance with the then current state of the art. It is intended that the scope of the invention be defined by the Claims as written and equivalents as applicable. Reference to a claim element in the singular is not intended to mean "one and only one" unless explicitly so stated. Moreover, no element, component, nor method or process step in this disclosure is intended to be dedicated to the public regardless of whether the element, component, or step is explicitly recited in the Claims. No claim element herein is to be construed under the provisions of 35 U.S.C. Sec. 112, sixth paragraph, unless the element is expressly recited using the phrase "means for. . ." and no method or process step herein is to be construed under those provisions unless the step, or steps, are expressly recited using the phrase "comprising the step(s) of. . .." What is claimed is: